## **Engenharia de Software**

A prática da informática teve início, em torno de 1950, quando os primeiros computadores foram criados.

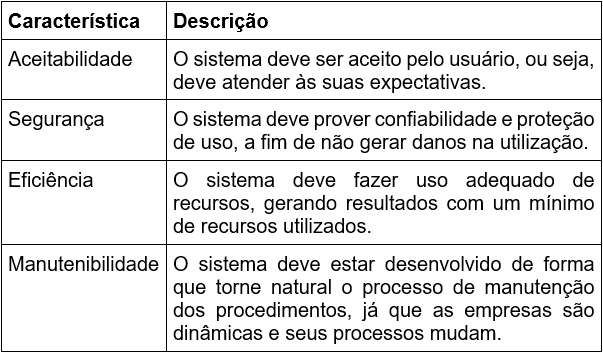
Em um primeiro momento, o incentivo maior era dado ao hardware, pois era importante disponibilizar a máquina com potencialidade suficiente para o processamento. O software era desenvolvido de acordo com a necessidade de aplicação. A necessidade de avanço do hardware era tão grande que postergou a necessidade por melhores softwares. Essa falta só seria sentida mais adiante quando a sobra de capacidade do hardware mostrasse o descompasso do estágio evolutivo do software, ou seja, a capacidade excessiva de hardware para um nível de software ainda incipiente.

Os softwares eram desenvolvidos sem o uso de técnicas formais e por profissionais ainda pouco preparados para essa “especialidade” e, considerando-se a especificidade inicial de aplicação dos computadores, ainda eram produtos absolutamente proprietários. Somente os programadores que os construíam tinham facilidade em manter os programas e entender a lógica preparada.

Os supostos analistas de sistemas não tinham habilidade interpessoal para estabelecer com os usuários uma relação de troca de conhecimento, tão importante na definição de sistemas, e não levavam em consideração as reais necessidades desses usuários. Acreditavam conhecer plenamente os processos, pois julgavam ser os melhores, mais sábios e espertos. Os usuários também contribuíram para a falta de relacionamento, pois se sentiam ameaçados pela chegada do computador e colocavam barreiras na exposição das necessidades, encarando a tecnologia como um inimigo e não uma ferramenta de trabalho. Eles escondiam as informações, boicotavam os sistemas e dificultavam a evolução da história. Estes fatos tinham como consequência a produção de sistemas inadequados à realidade, o que gerava insatisfação por parte do usuário, atraso no processo e retrabalho.

Diante desse panorama de desenvolvimento desordenado, insatisfação de usuário e excesso de retrabalho, estudos específicos levaram a criação da Engenharia de Software para ordenar o desenvolvimento de softwares mais funcionais e com melhor qualidade.

Segundo Sommerville (2019), as características essenciais para um bom software são: Aceitabilidade,Segurança, Eficiência e Manutenibilidade, descritos na Tabela 1.



Fonte: Sommerville, 2019 (adaptado)

Tabela 1 - Características essenciais de um bom software

A engenharia de software é o conjunto de técnicas, regras e métodos para desenvolvimento de software, buscando a qualidade não só em termos técnicos como funcionais. Além disso, consiste em definir uma metodologia com o faseamento em disciplinas e participação do usuário no processo de desenvolvimento, marcando o início do progresso ordenado de softwares.

**Mas o que são SOFTWARES?**

Softwares (Sistemas) representam um conjunto de funcionalidades (programas) que, juntos, estarão atendendo a um objetivo comum, através do processamento de informações. Vários são os tipos de softwares desenvolvidos: comerciais, industriais, jogos, automação, dentre outros. Fato é que, independente da proposição de atendimento do software, precisa-se especificar, desenvolver, validar e manter.

Para tanto, a Engenharia de Engenharia de Software definiu as Disciplinas e Ciclos de Vida necessários para execução do processo de desenvolvimento, utilizados de acordo com a natureza do software. Inclui-se ainda a visão sociotécnica neste processo, pois trará o conhecimento de todos os envolvidos na realização das atividades. Assuntos que serão tratados a seguir.

**DISCIPLINAS**

São atividades realizadas no desenvolvimento de sistemas para suprir a especificação, desenvolvimento, validação e manutenção, tais como: Levantamento de Requisitos, Análise, Projeto, Implementação, Teste, Implantação, Qualidade e Manutenção.

Especificação

Consiste no conhecimento do negócio e das regras aplicadas. Para isso são utilizadas as disciplinas denominadas por Levantamento de Requisitos e Análise.

**Levantamento de Requisitos** - disciplina que identifica as necessidades do negócio, definindo o escopo e conhecendo os usuários.

**Análise** - disciplina que conhece as regras de negócio, procedimentos e realiza a modelagem conceitual dos dados, necessários para realização de cada necessidade apontada na disciplina de Levantamento de Requisitos.

Desenvolvimento

Busca definir as partes físicas do sistema, definindo a estrutura dos bancos de dados e componentes que serão utilizados para realização das necessidades do sistema através das disciplinas Projeto e Implementação.

**Projeto** - Define a estrutura dos componentes (programas e funções), suas interações, desenho das interfaces e estrutura do modelo físico do banco de dados.

**Implementação** - Constrói os componentes na linguagem de programação definida no projeto.

Validação

Em todo processo de desenvolvimento são realizadas validações para que se possa obter a consistência das definições. As validações permeiam testes realizados em programas e verificam se os requisitos identificados e implementados estão de acordo com as especificações do usuário.

Esses processos de validação buscam a qualidade, que deve estar de encontro com as expectativas e desejos do usuário. As disciplinas de Engenharia de Software trabalhadas nesta etapa do processo de desenvolvimento são denominadas por **Teste** e **Qualidade**.

**IMPLANTAÇÃO**

Disciplina aplicada a fim de disponibilizar o software para uso do usuário.

Na implantação é preciso estar atento a dois procedimentos importantes: Treinamento e Carga dos dados.

Treinamento

Os usuários a trabalharem com o software a ser disponibilizado, devem conhecer os mecanismos desenvolvidos na operacionalização de suas necessidades. Para isso, deve-se organizar um treinamento e ambientalizar o usuário nas funções do software. Faz toda diferença para o sucesso do projeto!

Carga Dos Dados

Sabendo que os dados representam o coração e pulsar das empresas, temos a responsabilidade de utilizar os dados já armazenados e, com isso, devemos definir os procedimentos que estarão trazendo os dados para a nova estrutura criada, podendo, às vezes, requisitar procedimentos de conversão ou mesmo complemento de informações.

**MANUTENÇÃO**

Ajustes são desenvolvidos para adequação do software, ora já implantado no usuário. As manutenções acontecem por três motivos básicos: erro de programa, erro de requisitos ou novas necessidades.

Erro de Programa - problemas de programa identificados após a implantação do sistema, o que leva a correção.

Erro de Requisitos - inadequação no levantamento de requisitos diante das necessidades dos usuários.

Novas Necessidades - sabendo-se que a empresa é dinâmica em suas estratégias, novas operações surgem em seu processo de negócio e, consequentemente, provocam adequações nos softwares utilizados.

**GERÊNCIA DE PROJETO**

Em se tratando de um projeto de software, a necessidade da gerência de projeto também se torna importante, pois é preciso planejar e controlar as atividades desenvolvidas durante o desenvolvimento.

**CICLOS DE VIDA**

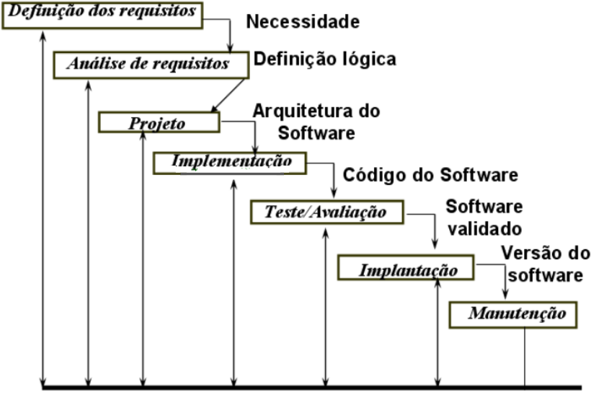
Os ciclos de vida representam o faseamento das atividades em etapas para compor o processo de desenvolvimento e quais os documentos devem ser gerados desde o conhecimento do problema até a sua solução instalada num computador. Muitos foram os ciclos definidos ao longo destes anos, para prototipagem rápida, prototipagem evolutiva, desenvolvimento convencional ou orientado a objetos. Porém, a abordagem a ser utilizada é a mesma: o conhecimento do problema, representação conceitual da solução e a representação de como a solução será instalada no computador. São eles: Cascata, Prototipagem, Espiral, Iterativo e Incremental, cada um com suas características próprias buscando, dentro da evolução, atender a deficiências.

**CICLO DE VIDA EM CASCATA**

O ciclo de vida em Cascata (Figura 1) foi o primeiro ciclo de vida proposto, definido na Metodologia estruturada, a partir da implementação dos conceitos da Engenharia de Software.

Características

* Documentos são gerados a cada fase e servem de entrada para a fase seguinte. Uma etapa só inicia ao término da anterior;
* Vulnerável para mudança de requisitos, pois é preciso retornar com o projeto completo;
* Longa duração para gerar resultado ao usuário.



Fonte: De autoria própria, 2022

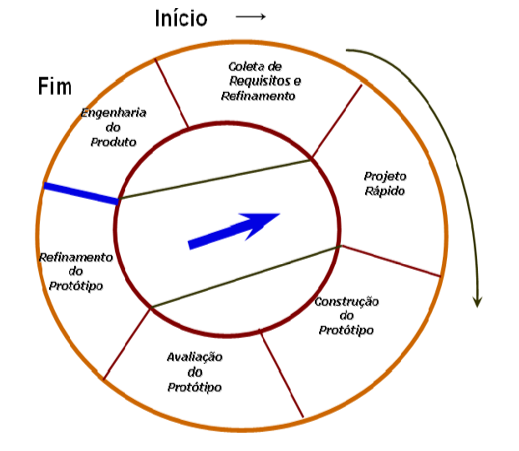
Figura 1: Ciclo de Vida em Cascata

**PROTOTIPAGEM**

O ciclo de vida Prototipagem (Figura 2) foi proposto na tentativa de minimizar o tempo de duração resultante do ciclo de vida em cascata, mas pela vulnerabilidade de produções incompletas, não foi adotado plenamente. Contudo, acabou tendo sua aplicação como técnica de validação/teste de software.

Características

* Desenvolvimento rápido para validação com o usuário. Interfaces são elaboradas e procedimentos desenvolvidos sem refinamentos, o que causa retrabalho ou implantações incompletas.



Fonte: De autoria própria, 2022.

Figura 2: Ciclo de Vida Prototipagem

✔ Protótipos são uma interessante solução para:

🡺 Sistemas novos - requisitos indefinidos;

🡺 Introduzir cultura do uso em uma organização;

🡺 Situações de emergência.

Obs: Desde que entendidas as limitações, protótipos podem ser uma boa ferramenta para experiências.

**O uso de Protótipos apresenta problemas**

✔ O usuário vê um protótipo como um sistema de funções importantes para ele, independente se o software que dá suporte a essas funções é frágil e difícil de manter;

✔ Para construir rapidamente um protótipo, o projetista lança mão do que está disponível, mesmo não sendo a “melhor” opção. Com o tempo ele se acostuma a essa solução e se esquece das deficiências;

✔ Portanto, há que se tomar cuidado com a utilização do enfoque para solução de problemas críticos.

**MODELO ESPIRAL**

O ciclo de vida Espiral (Figura 3), assim como o de Prototipagem foi proposto na tentativa de minimizar o tempo de duração resultante do ciclo de vida em cascata, mas por dificuldades tecnológicas de implementação não foi utilizado na totalidade.

Características

✔ Tentativa de união do ciclo tradicional com protótipos;

✔ Desenvolvimento em partes;

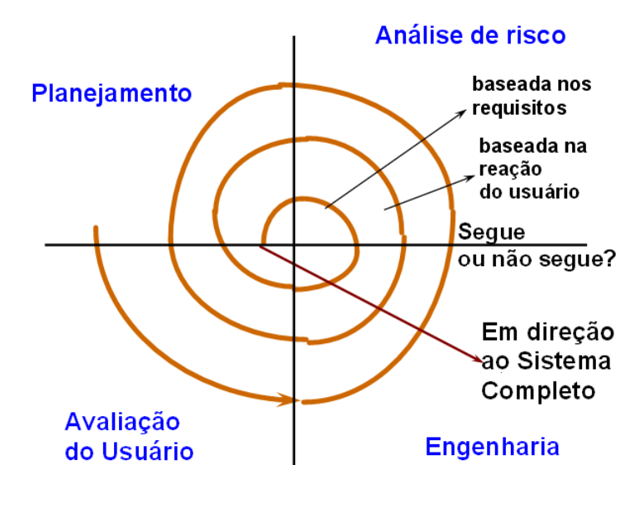
✔ Modelo realista, mas difícil de convencer a sua adoção;

✔ Pode ser incontrolável: sempre evolui e pode ser problemático se o centro da espiral não é o centro do sistema a ser desenvolvido;

✔ Depende muito de uma análise de riscos bem feita;

✔ Falta de cultura e conhecimento na adoção do modelo;

✔ Altamente dependente da Tecnologia.



Fonte: De autoria própria, 2022.

Figura 3: Ciclo de Vida Espiral

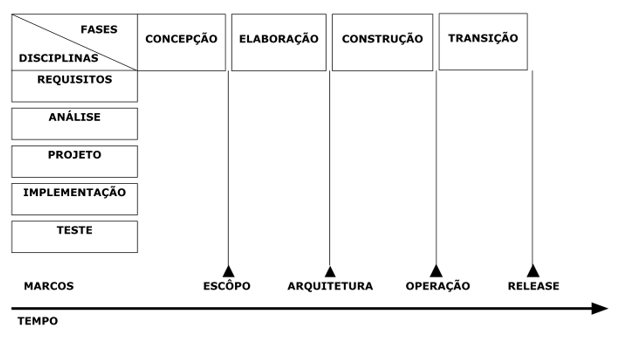
**ITERATIVO E INCREMENTAL**

O ciclo Iterativo e Incremental foi proposto na Metodologia Orientada a Objetos, aproveitando a ideia do ciclo de vida Espiral, divisão do desenvolvimento em partes, que não teve sua utilização em função da dependência da tecnologia, o que, com a orientação a objetos tornou-se possível em função da modularização e reutilização dos componentes.

Caracteriza-se por repetir o ciclo de vida para cada iteração e ao final incrementar a produção com uma nova versão do software. Uma iteração é representada pela execução do ciclo para a parte em desenvolvimento.

Características

* Desenvolvimento realizado em partes;
* Fácil para atender mudanças de requisitos, pois se a alteração afeta uma parte ainda não desenvolvida não tem repercussão e se afetar uma parte já desenvolvida os procedimentos são independentes;
* Usuário consegue visualizar resultados antecipadamente;
* Utiliza-se de quatro etapas: Concepção, Elaboração, Construção e Transição, onde utiliza-se das disciplinas em suas realizações conforme Figura 4.



Fonte: De autoria própria, 2022.

Figura 4: Ciclo de Vida Iterativo e Incremental

**FASES:**

**CONCEPÇÃO**

OBJETIVOS

* Definir necessidades reais dos patrocinadores;
* Delimitar claramente o escopo do projeto;
* Formular a arquitetura candidata;
* Levantamento de principais funcionalidades;
* A partir de um subconjunto chave de requisitos propor uma arquitetura;
* Planejamento: visão do projeto, estudo de viabilidade e análise de riscos;
* O patrocinador concorda com a realização de uma séria análise de projeto;

**ELABORAÇÃO**

OBJETIVOS

* capturar quase todos os casos de uso;
* estabelecer uma arquitetura sólida para guiar as fases de construção e transição;
* monitorar riscos e seu impacto no caso de negócio;
* refinar plano de projeto.
* planejamento: tarefas (por prioridade e risco de desenvolvimento) x divisão de responsabilidades x prazos individuais e duração de suas interações;
* montando a equipe;
* modificando o ambiente de implementação;
* estabelecer critério de avaliação;
* estender os requisitos e escolher Casos de Uso para participar da iteração;

**CONSTRUÇÃO**

OBJETIVOS

* Completar as realizações dos casos de uso, projetar as classes e subsistemas, implementá-los como componentes, fazendo testes individuais;
* O plano pode ser modificado por dois fatores: gap possível entre elaboração e construção; finanças e cronograma podem ser alterados;
* Alocação de recursos - Aumento significativo de pessoas;
* Definição do critério de avaliação;

**TRANSIÇÃO**

OBJETIVOS

* Atingir a capacidade final de operação:
* Modificar o produto para “aliviar” problemas não detectados nas fases anteriores;
* Corrigir defeitos;
* Garantir que o produto está pronto para ser entregue ao usuário;
* Realizar treinamentos;

**VISÃO SOCIOTÉCNICA**

A visão sociotécnica relacionada aos sistemas de informações se apresenta na medida em que a relação entre partes influenciadoras se tornam efetivas e necessárias para o sucesso do projeto. São considerados vários elementos do ambiente que, de alguma forma, influenciam o negócio abordado (Figura 5).



Fonte: De autoria própria, 2022

Figura 5: Visão SocioTécnica

Desta forma, compreendemos as bases iniciais para o desenvolvimento de um projeto de sucesso.

**Atividade Extra**

Para verificar a importância do planejamento, controle e organização no desenvolvimento de um projeto de software, assista ao vídeo “Empresa FazSite - Problemas processo de desenvolvimento de software” que pode ser encontrado na plataforma de vídeos (*Youtube*).

**Referência Bibliográfica**

SOMMERVILLE, I. **Engenharia de software**. 10.ed. São Paulo: Pearson Education do Brasil, 2018.